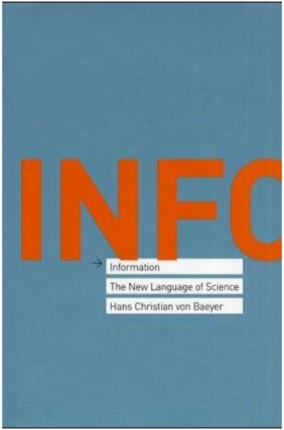
Three Concepts: Information

Teemu Roos

Complex Systems Computation Group
Department of Computer Science, University of Helsinki

Fall 2007



"Whether on the internet, encoded in radio waves or coursing through wires, information is all around us. Our senses record it, our brains process it and our genes pass it on. But what exactly is information? Can it be analysed and measured? In this extraordinary book, Hans Christian von Baeyer illuminates a concept that could soon become as central to science as space, time mass or energy."

1. Administrative issues
   - Course details
   - Prerequisites
   - What do I need to do?
   - Grading

1. Administrative issues
   - Course details
   - Prerequisites
   - What do I need to do?
   - Grading
2. What is Information?
   - Coding Game
   - Codes
   - Compression
   - Information

1. Administrative issues
   - Course details
   - Prerequisites
   - What do I need to do?
   - Grading
2. What is Information?
   - Coding Game
   - Codes
   - Compression
   - Information
3. What we will *not* talk about (except today)
   - *Reliable* communication
   - Error correcting codes
   - Repetition codes
   - Shannon's theorem

581286, Three Concepts: Information

- An advanced studies (laudatur) course.

581286, Three Concepts: Information

- An advanced studies (laudatur) course.
- Intelligent Systems sub-programme (optional).

581286, Three Concepts: Information

- An advanced studies (laudatur) course.
- Intelligent Systems sub-programme (optional).
- 6 credit units.

581286, Three Concepts: Information

- An advanced studies (laudatur) course.
- Intelligent Systems sub-programme (optional).
- 6 credit units.
- Period I, 5.9.–10.10.: Wed 13–16 B222.

581286, Three Concepts: Information

- An advanced studies (laudatur) course.
- Intelligent Systems sub-programme (optional).
- 6 credit units.
- Period I, 5.9.–10.10.: Wed 13–16 B222.
- Period II, 31.10.–5.12.: Wed 15–16 B222.

581286, Three Concepts: Information

- An advanced studies (laudatur) course.
- Intelligent Systems sub-programme (optional).
- 6 credit units.
- Period I, 5.9.–10.10.: Wed 13–16 B222.
- Period II, 31.10.–5.12.: Wed 15–16 B222.
- Instructor: **Teemu Roos**, A346,
  `teemu.roos at cs.helsinki.fi`
  Student tutoring hours: Thu 10:30-11:30AM (pls. make an appointment).

581286, Three Concepts: Information

- An advanced studies (laudatur) course.
- Intelligent Systems sub-programme (optional).
- 6 credit units.
- Period I, 5.9.–10.10.: Wed 13–16 B222.
- Period II, 31.10.–5.12.: Wed 15–16 B222.
- Instructor: **Teemu Roos**, A346,
  `teemu.roos at cs.helsinki.fi`
  Student tutoring hours: Thu 10:30-11:30AM (pls. make an appointment).
- Course assistant: **Jukka Perkiö**.
  `jukka.perkio at cs.helsinki.fi`

581286, Three Concepts: Information

- An advanced studies (laudatur) course.
- Intelligent Systems sub-programme (optional).
- 6 credit units.
- Period I, 5.9.–10.10.: Wed 13–16 B222.
- Period II, 31.10.–5.12.: Wed 15–16 B222.
- Instructor: **Teemu Roos**, A346,
  `teemu.roos at cs.helsinki.fi`
  Student tutoring hours: Thu 10:30-11:30AM (pls. make an appointment).
- Course assistant: **Jukka Perkiö**.
  `jukka.perkio at cs.helsinki.fi`
- `www.cs.helsinki.fi/group/cosco/Teaching/Information/2007/`

Prerequisites

No formal prerequisites **but** you will need

Prerequisites

No formal prerequisites **but** you will need

- Calculus: integrals, derivatives, convergence, ...

Prerequisites

No formal prerequisites **but** you will need

- Calculus: integrals, derivatives, convergence, ...
- Probability theory: joint & conditional distributions, expectations, law of large numbers, ...

Prerequisites

No formal prerequisites **but** you will need

- Calculus: integrals, derivatives, convergence, ...
- Probability theory: joint & conditional distributions, expectations, law of large numbers, ...
- Programming: language is up to you (but need to work in groups).

Prerequisites

No formal prerequisites **but** you will need

- Calculus: integrals, derivatives, convergence, ...
- Probability theory: joint & conditional distributions, expectations, law of large numbers, ...
- Programming: language is up to you (but need to work in groups).
- (Verbal) communication: individual and group presentations and written reports.

What do I need to do?

No regular exercises. Instead:

- Home assignments.

What do I need to do?

No regular exercises. Instead:

- Home assignments.
- Projects.

What do I need to do?

No regular exercises. Instead:

- Home assignments.
- Projects.
- Posters.

What do I need to do?

No regular exercises. Instead:

- Home assignments.
- Projects.
- Posters.
- Term paper.

What do I need to do?

No regular exercises. Instead:

- Home assignments.
- Projects.
- Posters.
- Term paper.

What do I need to do?

No regular exercises. Instead:

- Home assignments.
- Projects.
- Posters.
- Term paper.

You do *not* have to attend the classes, unless otherwise stated. However, we recommend that you do.

What do I need to do?

No regular exercises. Instead:

- Home assignments.
- Projects.
- Posters.
- Term paper.

You do *not* have to attend the classes, unless otherwise stated. However, we recommend that you do.

If you find that the course is not for you, please let us know *before* the project groups are formed. After that you are expected to stick with your group.

Grading

The course grading is based on:

1. Projects (50 %)

Grading

The course grading is based on:

1. Projects (50 %)
2. Poster (25 %)

Grading

The course grading is based on:

1. Projects (50 %)
2. Poster (25 %)
3. Term paper (25 %)

Grading

The course grading is based on:

1. Projects (50 %)
2. Poster (25 %)
3. Term paper (25 %)

In addition, home assignments are obligatory. You cannot pass the course without returning an acceptable solution.

Grading

The course grading is based on:

1. Projects (50 %)
2. Poster (25 %)
3. Term paper (25 %)

In addition, home assignments are obligatory. You cannot pass the course without returning an acceptable solution.

Late policy: 10 % of the available points are reduced for being late and for every 24h that your solution is late.

1. Administrative issues
   - Course details
   - Prerequisites
   - What do I need to do?
   - Grading
2. What is Information?
   - Coding Game
   - Codes
   - Compression
   - Information
3. What we will *not* talk about (except today)
   - *Reliable* communication
   - Error correcting codes
   - Repetition codes
   - Shannon's theorem

Coding Game

Form groups of 3–4 persons. Each group constructs a *code* for the letters A–Z by using as *code-words* unique sequences of dots • and dashes (—) like "•", "— •", " — • — —", etc.

A	________	G	________	M	________	S	________	Y	________
B	________	H	________	N	________	T	________	Z	________
C	________	I	________	O	________	U	________		
D	________	J	________	P	________	V	________		
E	________	K	________	Q	________	W	________		
F	________	L	________	R	________	X	________		

Coding Game

Use your code to *encode* the message
"`WHAT DOES THIS HAVE TO DO WITH INFORMATION`".

Coding Game

Use your code to *encode* the message
"`WHAT DOES THIS HAVE TO DO WITH INFORMATION`".

Now count how long the encoded message is using the rule:

- A dot •: 1 units.
- A dash —: 2 units.
- A space between words: 2 units.

Coding Game

Use your code to *encode* the message
"`WHAT DOES THIS HAVE TO DO WITH INFORMATION`".

Now count how long the encoded message is using the rule:

- A dot •: 1 units.
- A dash —: 2 units.
- A space between words: 2 units.

• • • — — — • • •: $1 + 1 + 1 + 2 + 2 + 2 + 1 + 1 + 1 = 10$.

Coding Game

Use your code to *encode* the message
“`WHAT DOES THIS HAVE TO DO WITH INFORMATION`”.

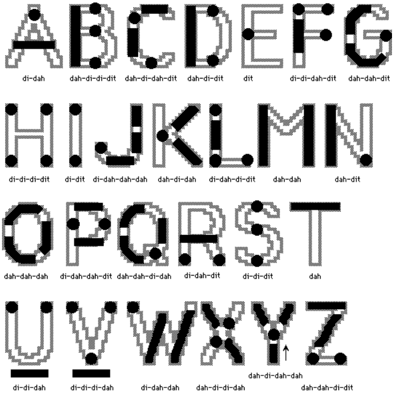
Now count how long the encoded message is using the rule:

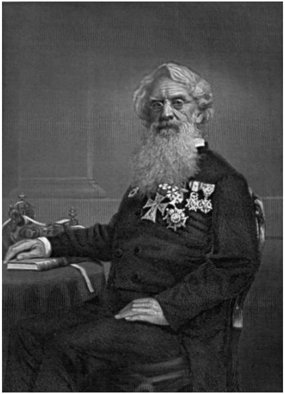
- A dot $\bullet$: 1 units.
- A dash —: 2 units.
- A space between words: 2 units.

$\bullet\bullet\bullet$ — — — $\bullet\bullet\bullet$: $1+1+1+2+2+2+1+1+1=10$.

The *coding rate* of your code is the length of the encoded message divided by the length of the original message, including spaces (42).

Coding Game





Samuel F.M. Morse (1791–1872)

Coding Game

WHAT DOES THIS HAVE TO DO WITH INFORMATION

Coding Game

```
WHAT DOES THIS HAVE TO DO WITH INFORMATION

.-- .... .- -   -.. --- . ...   - .... .. ...
.... .- ...- .   - ---   -.. ---   .-- .. - ....
.. -. ..-. --- .-. -- .- - .. --- -.
```

Coding Game

```
WHAT DOES THIS HAVE TO DO WITH INFORMATION
```

```
.-- .... .- -   -.. --- . ...   - .... .. ...
.... .- ...- .   - ---   -.. ---   .-- .. - ....
.. -. ..-. --- .-. -- .- - .. --- -.
```

51 dots, 36 dashes, 7 spaces: $51 + 72 + 14 = 137$ units.

Coding Game

```
WHAT DOES THIS HAVE TO DO WITH INFORMATION
```

```
.-- .... .- -   -.. --- . ...   - .... .. ...
.... .- ...- .   - ---   -.. ---   .-- .. - ....
.. -. ..-. --- .-. -- .- - .. --- -.
```

51 dots, 36 dashes, 7 spaces: $51 + 72 + 14 = 137$ units.

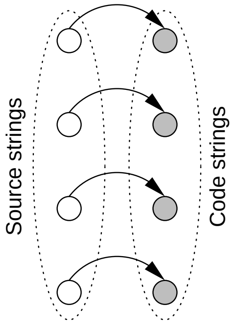
Morse code

Coding rate: $\dfrac{137}{42} \approx 3.26$

Did you do better or worse? Why?

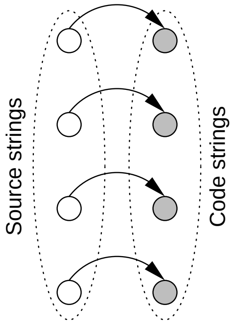
Codes

Lossless compression:
injective mapping

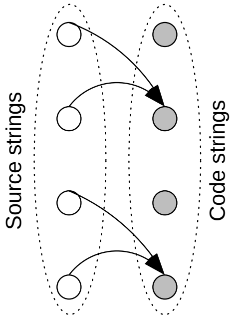


Codes

Lossless compression:
injective mapping

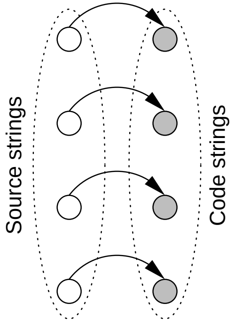


Lossy compression:
non-injective mapping

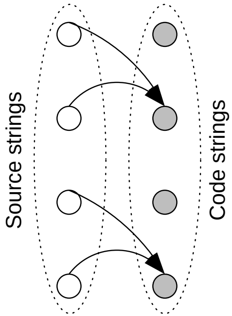


Codes

Lossless compression:
injective mapping



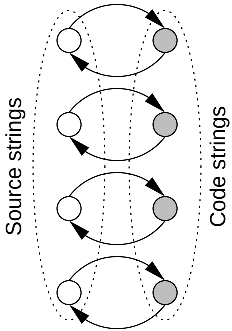
Lossy compression:
non-injective mapping



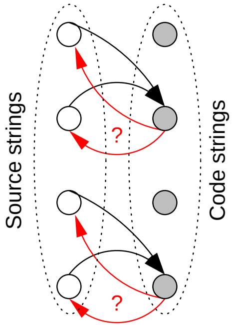
Only *lossless* codes are *uniquely decodable*.

Codes

Lossless compression:
injective mapping



Lossy compression:
non-injective mapping



Only *lossless* codes are *uniquely decodable*.

Examples

general purpose

`gzip`

`bzip`

Examples

general purpose	**gzip**
	bzip
image	**png**
	jpeg

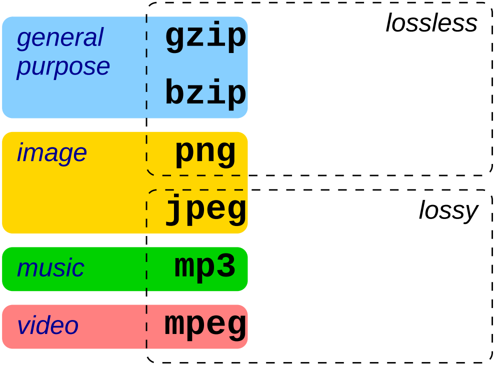
Examples

general purpose	**gzip**
	bzip
image	**png**
	jpeg
music	**mp3**

Examples

general purpose	**gzip**
	bzip
image	**png**
	jpeg
music	**mp3**
video	**mpeg**

Examples


lossless
general purpose
gzip
bzip
image
png
jpeg
lossy
music
mp3
video
mpeg

Examples

		compression ratio	
general purpose	**gzip**	~ 1 : 3	*lossless*
	bzip	~ 1 : 3.5	
image	**png**	~ 1 : 2.5	
	jpeg	~ 1 : 25	*lossy*
music	**mp3**	~ 1 : 12	
video	**mpeg**	~ 1 : 30	

Compression

Is it always possible to compress data?

Theorem

The proportion of binary strings compressible by more than k bits is at most 2^{-k}.

Compression

Is it always possible to compress data?

Theorem

The proportion of binary strings compressible by more than k bits is at most 2^{-k}.

Proof. For all $n \geq 1$, the number of binary strings of length n is 2^n. The number of binary code strings of length less than $n-k$ is $2^0 + 2^1 + 2^2 + \ldots + 2^{n-k-1} = 2^{n-k} - 1$. Thus the ratio is

$$\frac{2^{n-k}-1}{2^n} < \frac{2^{n-k}}{2^n} = 2^{-k}.$$

$\square$

Compression

Is it always possible to compress data?

Theorem

The proportion of binary strings compressible by more than k bits is at most 2^{-k}.

Proof. For all $n \geq 1$, the number of binary strings of length n is 2^n. The number of binary code strings of length less than $n-k$ is $2^0 + 2^1 + 2^2 + \ldots + 2^{n-k-1} = 2^{n-k} - 1$. Thus the ratio is

$$\frac{2^{n-k}-1}{2^n} < \frac{2^{n-k}}{2^n} = 2^{-k}.$$

$\square$

Less than 50 % of files are compressible by more than one bit.

Compression

Is it always possible to compress data?

Theorem

The proportion of binary strings compressible by more than k bits is at most 2^{-k}.

Proof. For all $n \geq 1$, the number of binary strings of length n is 2^n. The number of binary code strings of length less than $n-k$ is $2^0 + 2^1 + 2^2 + \ldots + 2^{n-k-1} = 2^{n-k} - 1$. Thus the ratio is

$$\frac{2^{n-k} - 1}{2^n} < \frac{2^{n-k}}{2^n} = 2^{-k}.$$

$\square$

Less than 1 % of files are compressible by more than 7 bits.

Compression

Is it always possible to compress data?

Theorem

The proportion of binary strings compressible by more than k bits is at most 2^{-k}.

Proof. For all $n \geq 1$, the number of binary strings of length n is 2^n. The number of binary code strings of length less than $n - k$ is $2^0 + 2^1 + 2^2 + \ldots + 2^{n-k-1} = 2^{n-k} - 1$. Thus the ratio is

$$\frac{2^{n-k} - 1}{2^n} < \frac{2^{n-k}}{2^n} = 2^{-k}.$$

$\square$

Less than 0.000000000000000000000000000001 % of files are compressible by 100 bits.

How is it possible?

Why was the compression ratio greater than one in all the examples we saw?

What are those rare files that are compressible?

Why are the files we use in practice so often compressible?

Compression

`echo <x> | gzip - | wc -c` # multiply by 8 for bits

Source string, x		$\ell(C(x))$	**ratio**
$aaa\ldots a$	$(10000 \times a)$	368	27.2 : 1.

Compression

`echo <x> | gzip - | wc -c` # multiply by 8 for bits

Source string, x		$\ell(C(x))$	**ratio**
$aaa \ldots a$	$(10000 \times a)$	368	$27.2 : 1.$
$aabaabbbbabbbbb\ldots$	(10000 random letters)	13456	$0.74 : 1$

Compression

`echo <x> | gzip - | wc -c` # multiply by 8 for bits

Source string, x		$\ell(C(x))$	**ratio**
$aaa\ldots a$	$(10000 \times a)$	368	$27.2 : 1.$
$aabaabbbbabbbbb\ldots$	(10000 random letters)	13456	$0.74 : 1$
$abababab\ldots ab$	$(5000 \times ab)$	368	$27.2 : 1$

Compression

`echo <x> | gzip - | wc -c` # multiply by 8 for bits

Source string, x		$\ell(C(x))$	**ratio**
$aaa\ldots a$	$(10000 \times a)$	368	$27.2 : 1.$
$aabaabbbbabbbbb\ldots$	(10000 random letters)	13456	$0.74 : 1$
$abababab\ldots ab$	$(5000 \times ab)$	368	$27.2 : 1$
$aaa\ldots abbb\ldots b$	$(5000 \times a, 5000 \times b)$	376	$26.6 : 1$

Compression

`echo <x> | gzip - | wc -c` # multiply by 8 for bits

Source string, x		$\ell(C(x))$	**ratio**
$aaa\ldots a$	$(10000 \times a)$	368	27.2 : 1.
$aabaabbbbabbbbb\ldots$	(10000 random letters)	13456	0.74 : 1
$abababab\ldots ab$	$(5000 \times ab)$	368	27.2 : 1
$aaa\ldots abbb\ldots b$	$(5000 \times a, 5000 \times b)$	376	26.6 : 1
$abbaababba\ldots$	$(1000 \times abbaababba)$	488	20.5 : 1

Compression

```
echo <x> | gzip - | wc -c
```
\# multiply by 8 for bits

Source string, x		$\ell(C(x))$	**ratio**
$aaa\ldots a$	$(10000 \times a)$	368	27.2 : 1.
$aabaabbbbabbbbb\ldots$	(10000 random letters)	13456	0.74 : 1
$abababab\ldots ab$	$(5000 \times ab)$	368	27.2 : 1
$aaa\ldots abbb\ldots b$	$(5000 \times a, 5000 \times b)$	376	26.6 : 1
$abbaababba\ldots$	$(1000 \times abbaababba)$	488	20.5 : 1

Strings following a rule are compressible?

Compression

`echo <x> | gzip - | wc -c` # multiply by 8 for bits

Source string, x		$\ell(C(x))$	**ratio**
$aaa\ldots a$	$(10000 \times a)$	368	$27.2 : 1.$
$aabaabbbbabbbbb\ldots$	(10000 random letters)	13456	$0.74 : 1$
$abababab\ldots ab$	$(5000 \times ab)$	368	$27.2 : 1$
$aaa\ldots abbb\ldots b$	$(5000 \times a, 5000 \times b)$	376	$26.6 : 1$
$abbaababba\ldots$	$(1000 \times abbaababba)$	488	$20.5 : 1$
$aaabbabbabb\ldots$	$(\pi$, 0–4 $\mapsto a$, 5–9 $\mapsto b)$	13416	$0.74 : 1$

π follows a rule but isn't compressed!

Compression

`echo <x> | gzip - | wc -c` # multiply by 8 for bits

Source string, x		$\ell(C(x))$	**ratio**
$aaa\ldots a$	$(10000 \times a)$	368	27.2 : 1.
$aabaabbbbabbbbb\ldots$	(10000 random letters)	13456	0.74 : 1
$abababab\ldots ab$	$(5000 \times ab)$	368	27.2 : 1
$aaa\ldots abbb\ldots b$	$(5000 \times a, 5000 \times b)$	376	26.6 : 1
$abbaababba\ldots$	$(1000 \times abbaababba)$	488	20.5 : 1
$aaabbabbabb\ldots$	$(\pi,\ 0\text{–}4 \mapsto a,\ 5\text{–}9 \mapsto b)$	13416	0.74 : 1

π follows a rule but isn't compressed!

Maybe it's just `gzip`? It would be possible to create to *special program* to compress π into a short file.

Compression

`echo <x> | gzip - | wc -c` # multiply by 8 for bits

Source string, x		$\ell(C(x))$	**ratio**
$aaa\ldots a$	$(10000 \times a)$	368	$27.2 : 1.$
$aabaabbbbabbbbb\ldots$	(10000 random letters)	13456	$0.74 : 1$
$abababab\ldots ab$	$(5000 \times ab)$	368	$27.2 : 1$
$aaa\ldots abbb\ldots b$	$(5000 \times a, 5000 \times b)$	376	$26.6 : 1$
$abbaababba\ldots$	$(1000 \times abbaababba)$	488	$20.5 : 1$
$aaabbabbabb\ldots$	(π, 0–4 $\mapsto a$, 5–9 $\mapsto b$)	13416	$0.74 : 1$

π follows a rule but isn't compressed!

Maybe it's just `gzip`? It would be possible to create to *special program* to compress π into a short file.

But what does it mean to compress an *individual* string???

Information

An individual string is "simple" (as opposed to "complex") if it can be compressed into a small file by a *prespecified* program.

Information

An individual string is "simple" (as opposed to "complex") if it can be compressed into a small file by a *prespecified* program.

But which program? `gzip` is not good for images (or for π).

Information

An individual string is “simple” (as opposed to “complex”) if it can be compressed into a small file by a *prespecified* program.

But which program? `gzip` is not good for images (or for π).

We can use several compressors if we prefix the code string by an index of the used program.

Information

An individual string is "simple" (as opposed to "complex") if it can be compressed into a small file by a *prespecified* program.

But which program? `gzip` is not good for images (or for π).

We can use several compressors if we prefix the code string by an index of the used program.

How about new compressors?

Information

An individual string is “simple” (as opposed to “complex”) if it can be compressed into a small file by a *prespecified* program.

But which program? `gzip` is not good for images (or for π).

We can use several compressors if we prefix the code string by an index of the used program.

How about new compressors? *Self-extracting files!*

Information

An individual string is "simple" (as opposed to "complex") if it can be compressed into a small file by a *prespecified* program.

But which program? `gzip` is not good for images (or for π).

We can use several compressors if we prefix the code string by an index of the used program.

How about new compressors? *Self-extracting files!*

Can it be made automatic? Find the shortest program to print x.

Information

An individual string is “simple” (as opposed to “complex”) if it can be compressed into a small file by a *prespecified* program.

But which program? `gzip` is not good for images (or for π).

We can use several compressors if we prefix the code string by an index of the used program.

How about new compressors? *Self-extracting files!*

Can it be made automatic? Find the shortest program to print x. **No.** *Kolmogorov complexity.*

Information

An individual string is "simple" (as opposed to "complex") if it can be compressed into a small file by a *prespecified* program.

But which program? `gzip` is not good for images (or for π).

We can use several compressors if we prefix the code string by an index of the used program.

How about new compressors? *Self-extracting files!*

Can it be made automatic? Find the shortest program to print x. **No.** *Kolmogorov complexity.*

Project II

Information vs. Complexity

Is complexity the same as information?

Information vs. Complexity

Is complexity the same as information?

Is there a lot of *information* in a random string? **No.**

Information vs. Complexity

Is complexity the same as information?

Is there a lot of *information* in a random string? **No.**

$$\begin{aligned} \textit{Complexity} &= \textit{Information} + \textit{Noise} \\ &= \textit{Regularity} + \textit{Randomness} \\ &= \textit{Algorithm} + \textit{Compressed file} \end{aligned}$$

1. Administrative issues
   - Course details
   - Prerequisites
   - What do I need to do?
   - Grading
2. What is Information?
   - Coding Game
   - Codes
   - Compression
   - Information
3. What we will *not* talk about (except today)
   - *Reliable* communication
   - Error correcting codes
   - Repetition codes
   - Shannon's theorem

Reliable communication

What if the code string was transmitter over a *noisy* connection?

Reliable communication

What if the code string was transmitter over a *noisy* connection?

- Modem line

Reliable communication

What if the code string was transmitter over a *noisy* connection?

- Modem line
- Satellite link

Reliable communication

What if the code string was transmitter over a *noisy* connection?

- Modem line
- Satellite link
- Hard disk

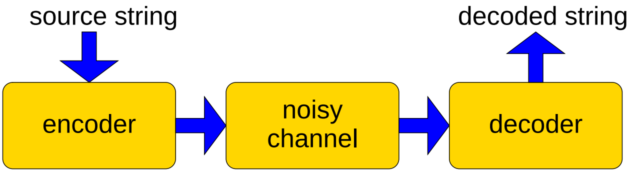
Reliable communication

What if the code string was transmitter over a *noisy* connection?

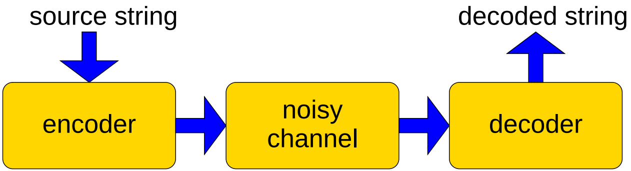
- Modem line
- Satellite link
- Hard disk

Can we recover the original message (without errors) from a noisy code string?

Error correcting codes



Error correcting codes



We want to minimize two things:

1. Length of the code string.
2. Probability of error.

Repetition codes

A simple idea: Just repeat the original string many times.

Repetition codes

A simple idea: Just repeat the original string many times.

`T R A N S M I S S I O N`

Repetition codes

A simple idea: Just repeat the original string many times.

`T R A N S M I S S I O N`

`TTTRRRAAANNNSSSMMMIIISSSSSSIIIOOONNN`

Repetition codes

A simple idea: Just repeat the original string many times.

`T R A N S M I S S I O N`

`TTTRRRAAANNNSSSMMMIIISSSSSSIIIOOONNN`

$\texttt{TTT}\underline{\texttt{H}}\texttt{RRAAANN}\underline{\texttt{B}}\texttt{SSSMMMIIISSSSS}\underline{\texttt{W}}\texttt{S}\underline{\texttt{P}}\texttt{I}\underline{\texttt{L}}\texttt{OOONN}\underline{\texttt{G}}$

Repetition codes

A simple idea: Just repeat the original string many times.

T R A N S M I S S I O N

TTTRRRAAANNNSSSMMMIIISSSSSSIIIOOONNN

TTTHRRAAANNBSSSMMMIIISSSSWSPILOOONNG

T R A N S M I S S ? O N

Repetition codes

A simple idea: Just repeat the original string many times.

`T R A N S M I S S I O N`

`TTTRRRAAANNNSSSMMMIIISSSSSSIIIOOONNN`

$$\texttt{TTT}\underline{\texttt{H}}\texttt{RRAAANN}\underline{\texttt{B}}\texttt{SSSMMMIIISSSS}\underline{\texttt{W}}\texttt{S}\underline{\texttt{P}}\texttt{I}\underline{\texttt{L}}\texttt{OOONN}\underline{\texttt{G}}$$

`T R A N S M I S S ? O N`

Transmission rate reduced to 1 : 3.

Repetition codes

A simple idea: Just repeat the original string many times.

`T R A N S M I S S I O N`

`TTTRRRAAANNNSSSMMMIIISSSSSSIIIOOONNN`

`TTTHRRAAANNBSSSMMMIIISSSSWSPILOOONNG`

`T R A N S M I S S ? O N`

Transmission rate reduced to 1 : 3.

If errors independent and symmetric, probability of error reduced to $\sim 3p^2$, where p is the error rate of the channel.

Shannon's Theorem

- Binary symmetric channel (BSC), error rate p:

$$\Pr[\hat{x} = 1 \mid x = 0] = \Pr[\hat{x} = 0 \mid x = 1] = p.$$

Shannon's Theorem

- Binary symmetric channel (BSC), error rate p:

$$\Pr[\hat{x} = 1 \mid x = 0] = \Pr[\hat{x} = 0 \mid x = 1] = p.$$

- *Channel capacity*

$$C(p) = 1 - H(p) = 1 - \left[p \log_2 \frac{1}{p} + (1-p) \log_2 \frac{1}{1-p} \right] .$$

Shannon's Theorem

- Binary symmetric channel (BSC), error rate p:

$$\Pr[\hat{x} = 1 \mid x = 0] = \Pr[\hat{x} = 0 \mid x = 1] = p.$$

- *Channel capacity*

$$C(p) = 1 - H(p) = 1 - \left[p \log_2 \frac{1}{p} + (1-p) \log_2 \frac{1}{1-p}\right] .$$

- For instance, $C(0.1) \approx 0.53$. Ratio about $1:2$.

Shannon's Theorem

- Binary symmetric channel (BSC), error rate p:

$$\Pr[\hat{x} = 1 \mid x = 0] = \Pr[\hat{x} = 0 \mid x = 1] = p.$$

- *Channel capacity*

$$C(p) = 1 - H(p) = 1 - \left[p \log_2 \frac{1}{p} + (1-p) \log_2 \frac{1}{1-p} \right] \quad .$$

- For instance, $C(0.1) \approx 0.53$. Ratio about $1:2$.

Noisy Channel Coding Theorem

For rates less than channel capacity, the error probability can be made arbitrarily small (but not zero).

Shannon's Theorem

Noisy Channel Coding Theorem

For rates less than channel capacity, the error probability can be made arbitrarily small (but not zero).

So what?

Shannon's Theorem

Noisy Channel Coding Theorem

For rates less than channel capacity, the error probability can be made arbitrarily small (but not zero).

Assume you want to transmit data with probability of error 10^{-15} over a BSC, $p = 0.1$. Using a repetition code, we need to transmit the message **59** times.
(You can check the math yourself. Hint: binomial distribution.)

Shannon's Theorem

Noisy Channel Coding Theorem

For rates less than channel capacity, the error probability can be made arbitrarily small (but not zero).

Assume you want to transmit data with probability of error 10^{-15} over a BSC, $p = 0.1$. Using a repetition code, we need to transmit the message **59** times.

Shannon's result says twice is enough.

Shannon's Theorem

Noisy Channel Coding Theorem

For rates less than channel capacity, the error probability can be made arbitrarily small (but not zero).

Assume you want to transmit data with probability of error 10^{-15} over a BSC, $p = 0.1$. Using a repetition code, we need to transmit the message **59** times.

If you want probability of error 10^{-100}, Shannon's result still says that twice is enough!

Shannon's Theorem

Noisy Channel Coding Theorem

For rates less than channel capacity, the error probability can be made arbitrarily small (but not zero).

Assume you want to transmit data with probability of error 10^{-15} over a BSC, $p = 0.1$. Using a repetition code, we need to transmit the message **59** times.

If you want probability of error 10^{-100}, Shannon's result still says that twice is enough!